САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ – ПРОЦЕССОВ УПРАВЛЕНИЯ

КАФЕДРА ИНФОРМАЦИОННЫХ СИСТЕМ

Смирнов Алексей Николаевич

Магистерская диссертация

Комбинированные алгоритмы синтеза управлений в некоторых классах управляемых систем

Направление магистратуры 01.04.02
«Прикладная математика и информатика»

Научный руководитель,
кандидат физ.-мат. наук, доцент
Еремин А. С.

Санкт-Петербург
2017

Содержание

Введение. Обзор литературы

Начиная примерно с середины XX века до настоящего времени математическая теория управления получила существенное развитие. Методы решения различных задач управления были разработаны такими выдающимися учеными как, Л. С. Понтрягин [1], Н. Н. Красовский [2], В. И. Зубов [3, 4], Р. Калман [5,6] и другими исследователями. Основными являются задачи программного управления, задачи стабилизации, а также проблемы оптимального управления и оптимизации систем управления. Специфика методов и подходов определяется сущностными свойствами объектов управления и соответствующим выбором математической модели (линейные или нелинейные, стационарные или нестационарные). При этом практика применения методов и реализация алгоритмов требуют учета специфики реальных условий эксплуатации разрабатываемых систем управления. Например, реализация только программных управлений возможна исключительно в идеальных условиях. Выясняется, что движение в реальной среде сопряжено с внешними возмущениями, которые невозможно учесть в рамках математической модели. Так возникает потребность в комбинировании методов программного управления и стабилизации программных движений по принципу обратной связи [3–8].

Еще одним из аспектов разработки методов синтеза позиционных управлений является возможность их реализации в режиме реального времени. Быстродействие современных бортовых компьютеров позволяет проектировать системы управления на альтернативных принципах, закладывать возможность расчета управляющих сигналов непосредственно в процессе движения объекта управления. В этом направлении существенных результатов добилась научная школа под руководством Р. Габасова и Ф. М. Кирилло-

вой [9–14]. Основная идея авторов данного подхода также связана с комбинированием различных алгоритмов для достижения конечной цели управления. Речь идет о сведении проблемы оптимального управления к интервальной задаче линейного программирования. Это оказывается возможным при дискретном характере управляющих сигналов. Различные приложения свидетельствуют о работоспособности данного подхода [12, 14–18].

Еще одним направлением разработки комбинированных алгоритмов управления являются подходы, связанные с построением вспомогательных линейных управляемых систем для более сложных по структуре нелинейных объектов. В некоторых случаях результаты построения управлений в линейных системах можно использовать для управления соответствующим нелинейным объектом.

К такому классу относятся, в частности, билинейные системы управления. Они широко используются для математического моделирования процессов в различных областях науки. Начиная с работ Р. Р. Мохлера (R. R. Mohler) [19–21], существует значительный интерес к использованию билинейных систем как математических моделей для описания поведения технических, экономических и биологических объектов. Кроме того, известны некоторые методы [22, 23] для аппроксимации нелинейных систем управления с помощью билинейных систем.

Билинейные системы по своей сути близки к линейным системам. Они возникают, когда параметры линейной модели могут быть изменены и рассматриваются в качестве управляющих параметров (см., например, [24, 25]). Это дает возможность использовать некоторые результаты математической теории линейных систем управления для задач синтеза управления в клас-

се билинейных систем. Это относится, прежде всего, к методам построения программных и стабилизирующих управлений [3–6], а также к задачам многопрограммного управления [26, 27].

В монографии [3] показано, что программное управление для линейной системы, переводящее ее в начало системы координат за конечное время, может быть представлено в эквивалентной форме позиционного управления. При этом замкнутая система имеет структуру, близкую к билинейной управляемой системе. В данной работе на основе этого свойства предложен метод синтеза управлений в билинейных однородных системах. Для его разработки потребовалось изучить предельные свойства позиционного управления для вспомогательной линейной системы, и построить соответствующий комбинированный алгоритм синтеза управления. В пакете MATLAB разработан набор компьютерных программ, которые протестированы на модельных примерах.

Постановка задачи

Изучить основные подходы к синтезу управлений в различных классах динамических управляемых систем. Проанализировать методы и алгоритмы синтеза управлений в линейных системах с целью возможного применения в некоторых нелинейных системах. Исследовать для этого варианты комбинирования стандартных алгоритмов.

Предположим, что объект управления описывается билинейной, однородной, управляемой системой дифференциальных уравнений

$$\dot{x} = P(t)x + Q(t)U(t)x,$$

где x — вектор фазового состояния размерности n; матрицы $P(t)$, $Q(t)$ размерностей $(n \times n)$, $(n \times r)$ имеют вещественные, непрерывные и ограниченные

при $t \geqslant 0$ элементы; $U(t)$ — $(r \times n)$-матрица управлений. Матрица $U(t)$ может иметь различную структуру в зависимости от конкретной прикладной задачи. Например, непосредственно все ее элементы могут быть управляемыми параметрами, либо представлять собой некоторые (линейные или нелинейные) функции заданного набора управляемых параметров, каждый из которых удовлетворяет своим ограничениям (см., например, [24]).

Задача управления состоит в том, чтобы перевести исходную систему из заданного фазового состояния $x = x_0$ при $t = 0$ в конечное состояние $x = 0$ при $t = T$. Требуется разработать алгоритм решения данной задачи и написать моделирующую компьютерную программу.

Основная проблема состоит в том, что позиционное управление, построенное для вспомогательной линейной системы, обладает определенными особенностями, а именно, матрица коэффициентов усиления по норме неограниченно возрастает при стремлении времени к финальной точке $t = T$. В связи с этим предлагается изучить ее свойства, а затем предложить комбинированный алгоритм синтеза управления. Идея в том, чтобы на первом этапе использовать позиционное управление, а на втором — стандартное программное.

Глава 1. Сравнительный анализ методов синтеза управлений

Сравнительный анализ идей методов синтеза управлений необходим для понимания области применимости каждого из них, а также учета нюансов реализации алгоритмов в конкретных прикладных задачах. В современной математической теории управления существует несколько «магистральных» направлений решения задач управления. Прежде всего, это сочетание в системе управления блока реализации программного режима (командного сигнала) и блока его стабилизации по принципу обратной связи [3–6].

Далее следует отметить методы позиционного управления, в основе которых лежит представление о том, что оптимальное программное управление в конкретной (возможно возмущенной) точке фазового пространства по своей сути можно трактовать как обратную связь по состоянию [10, 14]. Эта идея позволяет создавать алгоритмы решения задач управления в режиме реального времени для самых различных объектов [12, 14–18].

Наконец, В. И. Зубову принадлежит идея синтеза управления путем создания в замкнутой системе специальных особых точек (состояний), которые «затягивают» объект управления, заставляя его двигаться в нужном направлении [3]. Здесь необходимо специально отслеживать изменение нормы управляющего воздействия как основной характеристики, определяющей возможности практической реализации данного подхода. При этом, в случае ее допустимых значений, метод синтеза позволяет решать задачи управления в некоторых классах нелинейных моделей на основе вспомогательных линейных систем.

§ 1.1. Стабилизация программных движений по принципу обратной связи

Задача программного управления линейным объектом. Предположим, что объект управления описывается системой обыкновенных дифференциальных уравнений с управляемыми параметрами в правой части:

$$\dot{x} = P(t)x + Q(t)u + f(t), \tag{1.1}$$

где x — n-мерный вектор фазового состояния, u — r-мерный вектор управления, элементы $(n \times n)$- и $(n \times r)$-матриц $P(t)$, $Q(t)$ и векторной n-мерной функции $f(t)$ заданы при $t \geqslant 0$, вещественны, непрерывны и ограничены.

Задача программного управления [3]. Пусть в фазовом пространстве системы (1.1) заданы два множества M_0 и M_1. Они могут быть заданы как линейными, так и нелинейными соотношениями. Требуется построить управление $u(t)$, переводящее систему из начального состояния $x_0 \in M_0$ при $t = t_0$ в конечное состояние $x_1 \in M_1$ при $t = t_0 + T$, где t_0 — заданный момент времени, а T — горизонт решения задачи.

Перевод системы из начального заданного состояния x_0 при $t = t_0$ в конечное x_1 при $t = t_0 + T$ — это частный случай описанной постановки задачи.

Управление $u(t)$, решающее данную задачу, называется *программным*. Программные управления строятся в классе непрерывных или кусочно-непрерывных при $t \in [t_0, t_0 + T]$ функций.

Методы построения программных управлений в линейных системах основаны на возможности непосредственного интегрирования системы. Общее

решение записывается в форме Коши

$$x(t, t_0, x_0) = Y(t) \left(x_0 + \int_{t_0}^{t} Y^{-1}(\tau) \left(Q(\tau)u(\tau) + f(\tau) \right) d\tau \right),$$

где $Y(t)$ — фундаментальная матрица однородной системы $\dot{x} = P(t)x$. Затем это представление применяется в заданных ограничениях на состояния системы (множество M_1), что приводит к системе уравнений относительно управления $u(t)$.

В случае перевода системы в заданную точку x_1 при $t = t_0 + T$ имеем

$$x_1 = Y(t_0 + T) \left(x_0 + \int_{t_0}^{t_0+T} Y^{-1}(\tau) \left(Q(\tau)u(\tau) + f(\tau) \right) d\tau \right). \tag{1.2}$$

Данное интегральное уравнение, в общем случае, может быть решено методом неопределенных коэффициентов. Для этого искомое управление $u(t)$ представляют как линейную комбинацию наперед заданных функций, коэффициенты которой находят после подстановки представления в интегральное уравнение.

В [3] показано, что, если программное управление существует, то оно представимо в виде разложения по строкам матрицы $Y^{-1}(t)Q(t)$:

$$u(t) = \left(Y^{-1}(t)Q(t) \right)^T c + v(t), \tag{1.3}$$

где c — постоянный вектор неопределенных коэффициентов, $v(t)$ — векторная функция, удовлетворяющая условию ортогональности

$$\int_{t_0}^{t_0+T} Y^{-1}(\tau)Q(\tau)v(\tau)d\tau = 0.$$

После подстановки представления (1.3) в интегральное уравнение (1.2), получаем систему линейных алгебраических уравнений относительно искомого вектора c

$$Ac = b, \tag{1.4}$$

$$A = \int_{t_0}^{t_0+T} Y^{-1}(\tau)Q(\tau)\left(Y^{-1}(\tau)Q(\tau)\right)^T d\tau,$$

$$b = Y^{-1}(t_0+T)x_1 - x_0 - \int_{t_0}^{t_0+T} Y^{-1}(\tau)f(\tau)d\tau.$$

Отметим, что критерии управляемости пары состояний системы x_0, x_1, и полной управляемости линейной системы (для любых пар состояний x_0, x_1) формулируются в терминах рангов матриц A, $(A|b)$ и хорошо известны [3–6].

В заключение данного пункта заметим, что задачи программного управления нелинейными объектами или линейными, но при наличии нелинейных ограничений, описывающих множества начальных и конечных состояний, требуют серьезных усилий в плане применения численных методов решения краевых задач.

Задача стабилизации программных движений. Обозначим через $u_p(t)$ решение задачи программного управления. Если это управление подставить в исходную систему (1.1), то замкнутая система

$$\dot{x} = P(t)x + Q(t)u_p(t) + f(t)$$

будет иметь частное решение $x_p(t)$, обладающее свойством $x_p(t_0) = x_0$, $x_p(t_0+T) = x_1$. Решение $x_p(t)$ называется *программным решением* системы или *программным движением* объекта управления.

Задача стабилизации программных движений [3] состоит в обеспечении их асимптотической устойчивости по Ляпунову [28–30].

Для решения задачи стабилизации применяется принцип обратной связи. Опишем его идею. Вновь рассмотрим линейную систему (1.1), для которой построена пара функций $u_p(t)$, $x_p(t)$, т. е.

$$\dot{x}_p(t) \equiv P(t)x_p(t) + Q(t)u_p(t) + f(t).$$

Как принято в теории устойчивости движения при анализе устойчивости частных решений [30], сделаем в исходной системе замену переменных

$$y = x - x_p(t),$$

$$w = u - u_p(t),$$

где $y(t)$ — отклонение от программного движения, а w можно трактовать как дополнительное управление, способное обеспечить асимптотическую устойчивость частного решения $x_p(t)$. Действительно, $u = u_p(t) + w$.

В новых переменных, с учетом указанного тождества, получим линейную однородную систему в отклонениях

$$\dot{y} = P(t)y + Q(t)w. \tag{1.5}$$

Ее нулевое решение при нулевом управлении ($y = 0$, $w = 0$) соответствует программному движению исходной системы.

Принцип обратной связи по состоянию состоит в том, что управление, обеспечивающее асимптотическую устойчивость системы в отклонениях, строится в виде

$$w = C(t)y. \tag{1.6}$$

Оно зависит от отклонения y, которое измеряется датчиками системы наблюдения и считается известной величиной в процессе функционирования системы управления. Матрица коэффициентов усиления $C(t)$ вычисляется по специальным алгоритмам для стационарных [3, 4, 31] и нестационарных систем [8]. Конечная цель состоит в том, чтобы нулевое решение замкнутой системы (1.5), (1.6)

$$\dot{y} = (P(t) + Q(t)Q(t))y.$$

было асимптотически устойчиво.

Если задача стабилизации решена и управление $w = C(t)y$ построено, то для исходной системы управление вида

$$u = u_p(t) + C(t)(x - x_p(t)) \tag{1.7}$$

обеспечивает решение двух задач: 1) программного перевода в заданное состояние; 2) обеспечение асимптотической устойчивости (стабилизацию) программного движения.

Задача стабилизации с неполной информацией. В предыдущем пункте предполагалось, что весь вектор отклонений y доступен для измерений. Иногда это невозможно или слишком дорого (сбор и хранение информации в больших объемах, что характерно для больших экономических систем). Пусть теперь для измерения доступны только отдельные компоненты вектора y или их линейные комбинации. В этом случае к линейной системе добавляется уравнение измерителя

$$\begin{aligned} \dot{y} &= P(t)y + Q(t)w, \\ z &= R(t)y, \end{aligned} \tag{1.8}$$

где $(m \times n)$-матрица $R(t)$ и вектор измерений $z(t)$ считаются известными.

Задача стабилизации при неполной обратной связи [7]. Требуется построить такую оценку $\widehat{y}(t)$ вектора состояния $y(t)$, чтобы она обладала асимптотическим свойством

$$\widehat{y}(t) - y(t) \to 0 \quad \text{при} \quad t \to +\infty. \tag{1.9}$$

Если это возможно, то стабилизирующее управление конструируется в виде

$$w = C(t)\widehat{y}.$$

Система, которая формирует на выходе вектор $\widehat{y}(t)$ по измерениям $z(t)$ называется *асимптотическим идентификатором состояния.*

В [7] показано, что асимптотический идентификатор можно построить в виде

$$\dot{\widehat{y}} = P(t)\widehat{y} + Q(t)w + L(z - R(t)\widehat{y}), \tag{1.10}$$

где неизвестная $(n \times m)$-матрица L подлежит определению.

Идея данного представления основана на теореме о непрерывной зависимости решений систем ОДУ от их правых частей [32]. Действительно, слагаемое $L(z - R(t)\widehat{y})$ учитывает качество оценки состояния. При этом с учетом $z - R(t)\widehat{y} = R(t)(y - \widehat{y})$ и в идеальной ситуации при $y(t) \equiv \widehat{y}(t)$, уравнение (1.10) эквивалентно системе в отклонениях (1.8) относительно $y(t)$ с точностью до обозначений.

Таким образом, задача сводится к выбору матрицы L так, чтобы имело место асимптотическое свойство оценки (1.9) и существовало стабилизирующее управление $w = C(t)\widehat{y}$.

Общий комбинированный алгоритм построения и стабилизации программных движений состоит из следующих шагов:

1. Построить фундаментальную матрицу $Y(t)$ однородной системы $\dot{x} = Px$ и найти обратную к ней $Y^{-1}(t)$.

2. Построить матрицу и вектор коэффициентов системы (1.4):

$$A = \int_{t_0}^{t_0+T} Y^{-1}(\tau)Q\left(Y^{-1}(\tau)Q\right)^T d\tau,$$

$$b = Y^{-1}(t_0+T)x_1 - x_0 - \int_{t_0}^{t_0+T} Y^{-1}(\tau)f(\tau)d\tau.$$

3. Найти решение системы алгебраических уравнений (1.4) $Ac = b$.

4. Построить программное управление

$$u(t) = \left(Y^{-1}(t)Q\right)^T c.$$

5. Задать эталонные собственные числа. Построить матрицу коэффициентов усиления стабилизирующего управления $w = Cy$.

6. Для случая неполной обратной связи дополнительно найти матрицу коэффициентов L асимптотического идентификатора (1.10).

7. Сформировать комбинированное управление (1.7):

$$u = u_p(t) + C(x - x_p(t))$$

и провести тестирование программы.

Численная реализация. Для реализации общего комбинированного алгоритма программного управления и стабилизации линейной системы написана программа в пакете MATLAB (см. Приложение 1). Эта программа реализует случай линейных стационарных систем и состоит из двух основных блоков: блока расчета программного управления и блока стабилизации. Блок программного управления реализует пункты 1–4 общего алгоритма, а блок стабилизации — пункты 5–7.

Для иллюстрации работы и тестирования программы рассмотрим линейную стационарную систему (1.1) вида

$$\dot{x} = \begin{pmatrix} -5 & -5 \\ 3 & -1 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u + 0{,}1 \begin{pmatrix} \sin t \\ \cos t \end{pmatrix},$$

а также исходные данные задачи программного управления

$$x_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad x(T) = \begin{pmatrix} 3 \\ 4 \end{pmatrix}, \quad t_0 = 0, \quad T = 20.$$

На рис. 1, 2 представлены интегральные кривые системы, замкнутой программным управлением.

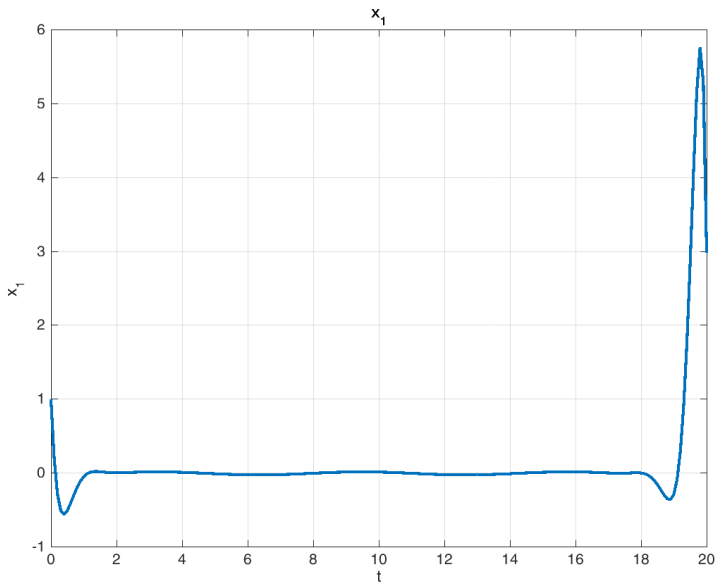

Рис. 1. Компонента x_1 программного движения

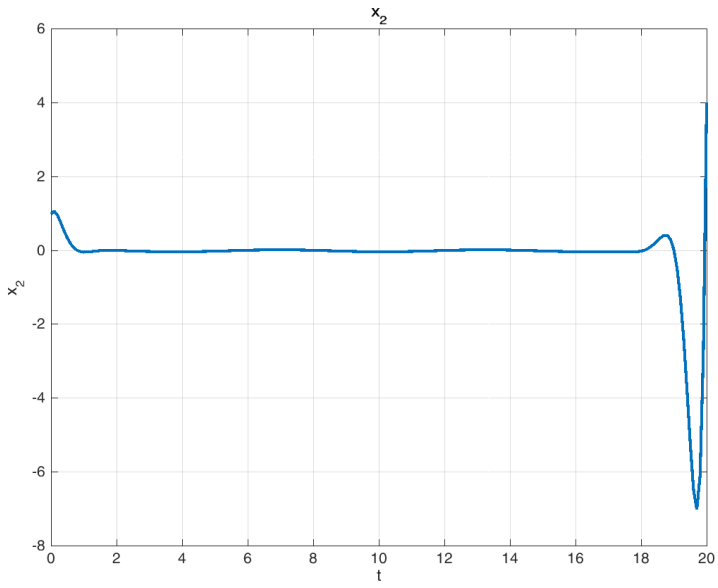

Рис. 2. Компонента x_2 программного движения

Для иллюстрации работы блока стабилизации движения система была подвергнута двум возмущениям. В начальный момент времени на величину 0,5 по компонентам x_1, x_2, а затем в момент времени $t = 8$ на величину 1 также по обеим компонентам. Результаты работы блока стабилизации представлены на рис. 3, 4.

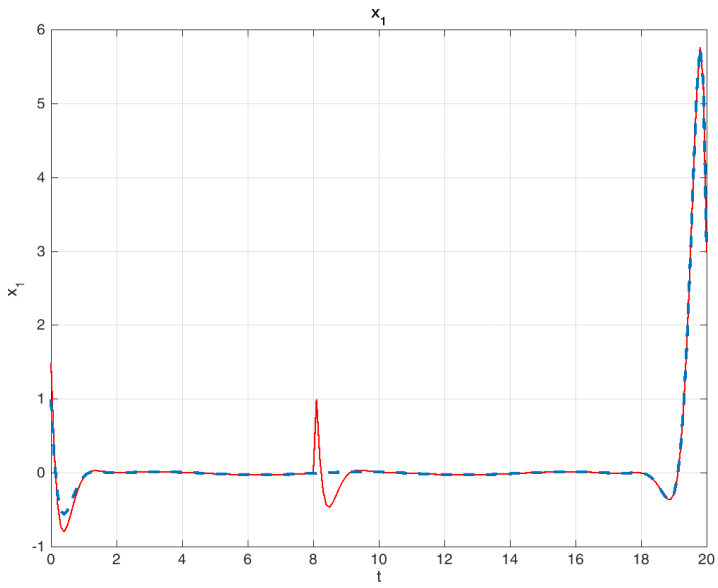

Рис. 3. Пунктир — программное движение по компоненте x_1, сплошная линия — движение с учетом стабилизации

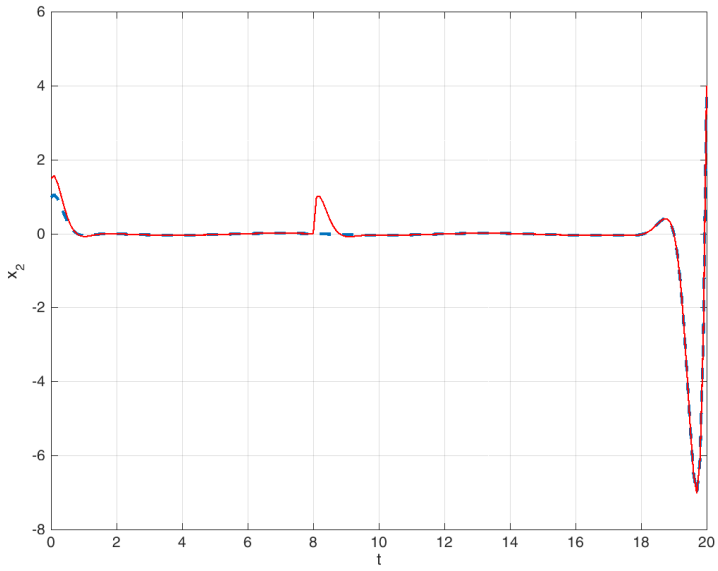

Рис. 4. Пунктир — программное движение по компоненте x_2, сплошная линия — движение с учетом стабилизации

В заключении данного параграфа отметим основную особенность разработанного алгоритма. В нем все основные элементы системы управления (программное и стабилизирующее управления) рассчитываются заранее, до начала движения объекта управления, и в дальнейшем уже не могут быть изменены. Это обстоятельство накладывает существенные ограничения на область применения данного подхода.

§ 1.2. Метод позиционной оптимизации Габасова

Сведение проблемы оптимального программного управления к задаче линейного программирования. Рассмотрим задачу программного управления линейным объектом с учетом линейной целевой функции [14]:

$$\begin{gathered} \dot{x} = P(t)x + Q(t)u, \\ x(t_0) = x_0, \quad Hx(t_0+T) = g, \quad |u(t)| \leqslant l, \\ \gamma^T x(t_0+T) \to \max_u . \end{gathered} \tag{1.11}$$

Здесь множество конечных состояний M_1 совпадает с линейным многообразием $Hx(t_0+T) = g$, где H — заданная $(m \times n)$-матрица, $\operatorname{rank} H = m$, g — заданный постоянный вектор, γ^T — постоянная строка коэффициентов целевой функции, константа l — ограничение на управление $u(t)$, которое считается скалярным и принадлежит классу кусочно-постоянных функций:

$$u(t) = u(t_0 + (k-1)h) = u_k, \quad t \in [t_0 + (k-1)h,\ t_0 + kh), \quad k = \overline{1, N},$$

где N — число точек разбиения интервала T на элементарные отрезки длиной $h = T/N$.

Запишем формулу Коши общего решения управляемой системы ОДУ при $t = t_0 + T$

$$x(t_0+T) = Y(t_0+T)\left(x_0 + \int_{t_0}^{t_0+T} Y^{-1}(\tau)Q(\tau)u(\tau)d\tau\right).$$

Тогда целевая функция примет вид

$$\gamma^T Y(t_0+T) \int_{t_0}^{t_0+T} Y^{-1}(\tau)Q(\tau)u(\tau)d\tau \longrightarrow \max_u .$$

Слагаемое $\gamma^T Y(t_0+T)x_0$ удалено, так как оно есть константа и не влияет на поиск оптимального управления.

С учетом дискретности управления получаем

$$\gamma^T Y(t_0+T)\sum_{k=1}^{N}\int_{t_0+(k-1)h}^{t_0+kh} Y^{-1}(\tau)Q(\tau)d\tau u_k \longrightarrow \max_u .$$

Используя формулу Коши в представлении для многообразия, описывающего возможное расположение финального состояния, получим систему уравнений относительно u_k, $k=\overline{1,N}$:

$$HY(t_0+T)\sum_{k=1}^{N}\int_{t_0+(k-1)h}^{t_0+kh} Y^{-1}(\tau)Q(\tau)d\tau u_k = g - HY(t_0+T)x_0.$$

При вычислении интегралов в двух последних формулах нужны значения обратной матрицы $Y^{-1}(t)$ в соответствующие моменты времени. Для линейных стационарных систем их можно вычислять, используя известное свойство $Y^{-1}(t)=Y(-t)$. В случае линейных нестационарных систем удобно использовать фундаментальную матрицу сопряженной системы [14, 32].

Введем обозначения:

$$U=(u_1,\ldots,u_N)^T,$$

$$\Gamma=(\Gamma_1,\ldots,\Gamma_N),\quad \Gamma_k=\gamma^T Y(t_0+T)\int_{t_0+(k-1)h}^{t_0+kh} Y^{-1}(\tau)Q(\tau)d\tau,$$

$$B=(B_1,\ldots,B_N),\quad B_k=HY(t_0+T)\int_{t_0+(k-1)h}^{t_0+kh} Y^{-1}(\tau)Q(\tau)d\tau,$$

$$\widehat{g}=g-HY(t_0+T)x_0,$$

Здесь B — $(m\times N)$-матрица, U — вектор значений управления, Γ — строка, $k=\overline{1,N}$.

В результате исходная динамическая задача оптимального программного управления (1.11) сводится к следующей задаче линейного программирования:

$$\Gamma U \to \max_u,$$
$$BU = \widehat{g}, \tag{1.12}$$
$$-l \leqslant u_k \leqslant l, \quad k = \overline{1, N}.$$

Адаптивный метод Габасова. Запишем интервальную задачу линейного программирования (ИЗЛП) в стандартной форме [11]:

$$c^T x \to \max_u,$$
$$b_* \leqslant Ax \leqslant b^*, \tag{1.13}$$
$$d_* \leqslant x \leqslant d^*.$$

Здесь $c^T = (c_1, \ldots, c_n)$ — строка коэффициентов целевой функции, A — $(m \times n)$-матрица затрат, $x = (x_1, \ldots, x_n)^T$ — вектор планов, $z = Ax$ — вектор затрат. Соотношения $b_* \leqslant Ax \leqslant b^*$ называются основными ограничениями, $d_* \leqslant x \leqslant d^*$ — прямыми.

В предыдущем пункте было показано, что задача оптимального программного управления (1.11) сводится к ИЗЛП (1.12). При этом есть важные нюансы. А именно, матрица затрат B имеет размеры $(m \times N)$. При большом числе интервалов разбиения N (сотни или тысячи единиц) эта матрица имеет «ленточную» структуру, т. е. число строк гораздо меньше числа столбцов. Тогда ограничения в ИЗЛП образуют симплекс соответствующей структуры, что является проблемой для классических методов решения задач линейного программирования (например, симплекс-метода).

Научная школа профессора Р. Габасова (он сам и его ученики) разработали и внедрили в практику решения задач оптимального управления специальный подход — *адаптивный метод*, ориентированный на учет указанно-

го нюанса и создали на его основе быстродействующие алгоритмы решения ИЗЛП. Опишем суть и основы адаптивного метода, следуя работе [11].

Пусть $I = 1, \ldots, m$ и $J = 1, \ldots, n$ – множества индексов строк и столбцов матрицы основных ограничений A. Совокупность подмножеств $I_{op} \in I$ и $J_{op} \in J$ назовем опорным множеством или «опорой» и обозначим через $K_{op} = \{I_{op}, J_{op}\}$, при условии, что $|I_{op}| = |J_{op}|$ и $\det A(I_{op}, J_{op}) \neq 0$.

Пару элементов $\{x, K_{op}\}$ принято называть *опорным планом*. Для решения ИЗЛП (1.13) нужно найти оптимальный план, на котором целевая функция достигает максимума.

Опорный план называется невырожденным, если $b_{*no} < z < b^*_{no}$ и $d_{*op} \leqslant x \leqslant d^*_{op}$, т. е. неопорные компоненты вектора затрат и опорные компоненты плана не выходят на границу симплекса.

Любой опоре $K_{op} = \{I_{op}, J_{op}\}$ соответствуют вектор потенциалов $u(I)$ и вектор оценок $\Delta(J)$, заданные формулами

$$\begin{cases} u(I_{op}) = \left(A(I_{op}, J_{op})^{-1}\right)^T c(J_{op}), \\ u(I_{no}) = 0, \end{cases}$$

$$\begin{cases} \Delta(J_{op}) = 0, \\ \Delta(J_{no}) = c(J_{no}) - A(I_{op}, J_{no})^T u(I_{op}). \end{cases}$$

Введенные термины позволяют сформулировать следующий критерий оптимальности.

Теорема 1 [11]. *Для оптимальности плана x достаточно существования такой опоры K_{op}, при которой на опорном плане $\{x, K_{op}\}$ выполняются соотношения*

$$\begin{cases} u_i \leqslant 0, & \textit{если } z_i = b_{*i}, \\ u_i \geqslant 0, & \textit{если } z_i = b_i^*, \\ u_i = 0, & \textit{если } b_{*i} < z_i < b_i^*,\ i \in I_{op}; \end{cases}$$

$$\begin{cases} \Delta_j \leqslant 0, & \textit{если } x_j = d_{*j}, \\ \Delta_j \geqslant 0, & \textit{если } x_j = d_j^*, \\ \Delta_j = 0, & \textit{если } d_{*j} < x_j < d_j^*,\ j \in J_{no}. \end{cases}$$

Обратно, если x — оптимальный план, и при некоторой опоре K_{op} пара $\{x, K_{op}\}$ — невырожденный опорный план, то на этом плане выполняются указанные соотношения.

Критерий оптимальности основан на том, что знаки величин u_i и Δ_j, которые определяют скорость изменения целевой функции, должны быть такими, чтобы в точке максимума допустимые вариации независимых переменных не приводили к росту значения целевой функции.

Общий алгоритм адаптивного метода. В подходе, предложенном Р. Габасовым, опора никак не зависит от текущего плана. Поэтому их можно менять независимо друг от друга, что позволяет эффективно строить оптимальный план. Благодаря гибкости опоры (вариации опорных и неопорных компонент), метод хорошо справляется с задачами, в которых количество переменных существенно превосходит число основных ограничений (как в задаче оптимального программного управления).

Метод включает в себя две фазы:

1. Построение начального опорного плана $\{x_1, K_{op}^1\}$ (первого приближения).
2. Построение последующих приближений $\{x_k, K_{op}^k\} \quad \to \quad \{x_{k+1}, K_{op}^{k+1}\}$, $k = 1, 2, \ldots$

В основу метода положен принцип уменьшения оценки приращения целевой функции. Для повышения эффективности на каждом его шаге опора строится заново. Поэтому вторая фаза состоит из алгоритма замены плана x_k и алгоритма замены опоры K_{op}^k. Эти алгоритмы описаны в [11] достаточно подробно, там же приведены описания блоков моделирующих компьютерных программ. В работах [12,14–18] представлены примеры конкретных приложений адаптивного метода в задачах оптимального управления.

Синтез оптимальной обратной связи на основе адаптивного метода. Вернемся к задаче (1.11) и предположим, что теперь на систему действуют возмущения $f(t)$. Поскольку возмущение действует постоянно, то задачу (1.11) будем рассматривать для произвольного начального момента τ и на соответствующем интервале времени $t \in [\tau, t_0 + T]$:

$$\begin{gathered} \dot{x} = P(t)x + Q(t)u + f(t), \\ x(\tau) = z, \quad Hx(t_0 + T) = g, \quad |u(t)| \leqslant l, \\ \gamma^T x(t_0 + T) \to \max_u . \end{gathered} \tag{1.14}$$

При наличии возмущений необходимо решать задачу стабилизации программного режима. Покажем основную идею метода Габасова по формированию обратной связи. Считаем, что управление из класса кусочно-постоянных функций. Пусть $W(\tau)$ — множество допустимых управлений. Обозначим $u^0(t|(\tau, z)) \in W(\tau)$, $t \in [\tau, t_0 + T]$ — оптимальное управление задачи (1.14) для (t, z), $z \in X(\tau)$, $X(\tau)$ — множество состояний системы в момент τ, для которых задача (1.14) имеет решение при фиксированном τ.

Как было показано выше, адаптивный метод позволяет для моментов времени $t_0, t_0 + h, \ldots, \tau$ построить соответствующие управления $u^*(t_0)$, $u^*(t_0 + h), \ldots, u^*(\tau)$. Тогда следующее фактическое состояние системы

$x^*(\tau+h)$, в котором она оказалась через еще один интервал h, связано с состоянием $x^0(\tau+h)$ в момент $\tau+h$ при отсутствии возмущения следующим соотношением:

$$x^*(\tau+h) = x^0(\tau+h) + \int_\tau^{\tau+h} Y(\tau+h)Y^{-1}(\gamma)f(\gamma)\ d\gamma. \qquad (1.15)$$

Тогда в момент $\tau+h$ надо решить следующую задачу оптимального управления

$$\dot{x} = P(t)x + Q(t)u + f(t),$$

$$x(\tau+h) = x^*(\tau+h), \quad Hx(t_0+T) = g, \quad |u(t)| \leqslant l, \quad t \in [\tau+h, t_0+T],$$

$$\gamma^T x(t_0+T) \to \max_u .$$

Очевидно, что эта задача может быть сведена к ИЗЛП указанным выше методом и решена при помощи адаптивного метода.

Осталось сделать главный вывод. Пусть $u^0\big(\tau+h|(\tau+h, x^*(\tau+h))\big)$ — оптимальное решение данной задачи. Поскольку оно зависит от текущего возмущенного состояния системы $x^*(\tau+h)$, то оно представляет собой позиционное стабилизирующее управление (один из видов динамической обратной связи).

Заметим, что при тестировании компьютерных программ состояние $x^*(\tau+h)$ можно имитировать на каждом шаге по формуле (1.15), а в реальных условиях эксплуатации объекта управления оно должно определяться датчиками системы позиционирования (наблюдения).

§ 1.3. Выводы по главе 1

В данной главе рассмотрены два различных подхода к проблеме синтеза управлений в линейных системах. Дадим их краткую сравнительную характеристику.

Комбинирование программных и стабилизирующих управлений в целом решает поставленную задачу устойчивого перевода объекта управления в заданное состояние. Отличительная особенность метода — все вычисления делаются заранее, до начала движения. Система управления проектируется однозначно только на отработку конкретного программного режима. Это существенно сужает возможности ее применения. В некотором смысле такая система управления является «одноразовой»: один программный режим, один тип стабилизатора. На практике может возникнуть возмущение, по отношению к которому ресурс системы стабилизации, детерминированный набором эталонных собственных чисел, либо избыточен, либо недостаточен.

Указанные недостатки первого подхода отчасти решает метод позиционной оптимизации. Изначальное дискретное представление допустимых управлений, ориентация на их перерасчет в режиме реального времени с учетом реальных (неучтенных заранее) возмущений, расширяют возможности метода в плане приложений и его самонастройки на условия эксплуатации. К недостаткам можно отнести необходимость на каждом шаге решать задачу линейного программирования большой размерности, что требует серьезных вычислительных и прочих ресурсов бортовой системы управления, например, наличия датчиков высокоточного позиционирования.

Отметим, что оба описанных подхода нельзя непосредственно перенести на классы нелинейных систем, требуется их существенная модификация. В главе 2 рассмотрен альтернативный принцип синтеза управлений в линейных системах, обладающий преимуществом в этом смысле. А именно, замкнутая линейная система имеет структуру, близкую к классу билинейных систем, что открывает дополнительные возможности применения данного подхода.

Глава 2. Синтез управлений в линейных системах

В этом разделе кратко описан метод синтеза управлений в линейных системах В. И. Зубова [3], а также рассмотрены вопросы его численной реализации. Это необходимо для дальнейшего изложения принципов построения управлений в классе билинейных управляемых систем.

§ 2.1. Линейная нестационарная система

Рассмотрим вспомогательную линейную систему

$$\dot{x} = P(t)x + Q(t)u + f(t), \tag{2.1}$$

где x — n-мерный вектор фазового состояния, u — r-мерный вектор управления, элементы $(n \times n)$- и $(n \times r)$-матриц $P(t)$, $Q(t)$ и векторной n-мерной функции $f(t)$ заданы при $t \geqslant 0$, вещественны, непрерывны и ограничены.

Программное управление для линейной системы (2.1), переводящее ее из начального положения $x(t_0) = x_0$ в начало системы координат $x(t_0 + T) = 0$ за конечное время T, может быть представлено в эквивалентной форме позиционного управления

$$u = u(t, x). \tag{2.2}$$

Покажем, как построить управление (2.2). Положим $t_0 = 0$ и запишем формулу Коши общего решения системы (2.1):

$$x(t, 0, x_0) = Y(t) \left(x_0 + \int_0^t Y^{-1}(\tau) \left(Q(\tau)u(\tau) + f(\tau) \right) d\tau \right),$$

где $Y(t)$ — фундаментальная матрица однородной системы $\dot{x} = P(t)x$, нормированная в нуле: $Y(0) = E$. Учитывая граничное условие из постановки

задачи $x(T) = 0$, получим интегральное уравнение относительно $u(t)$:

$$0 = Y(T)\left(x_0 + \int_0^T Y^{-1}(\tau)\left(Q(\tau)u(\tau) + f(\tau)\right)d\tau\right). \tag{2.3}$$

Введем обозначения

$$B(t) = Y^{-1}(t)Q(t), \quad A(t,T) = \int_t^T B(\tau)B^T(\tau)d\tau, \quad A(0,T) = \int_0^T B(\tau)B^T(\tau)d\tau$$

и будем искать решение интегрального уравнения (2.3) в виде

$$u(t) = B^T(t)c + v(t), \tag{2.4}$$

где c — постоянный вектор, $v(t)$ — векторная функция, удовлетворяющая условию ортогональности

$$\int_0^T B(\tau)v(\tau)d\tau = 0. \tag{2.5}$$

Подставляя представление (2.4) в интегральное уравнение (2.3), учитывая (2.5) и введенные обозначения, получим

$$-x_0 = A(0,T)c + \int_0^T Y^{-1}(\tau)f(\tau)d\tau. \tag{2.6}$$

Предположим, что матрица $A(0,T)$ неособая, тогда из (2.6) получим

$$c = -A^{-1}(0,T)\left(x_0 + \int_0^T Y^{-1}(\tau)f(\tau)d\tau\right). \tag{2.7}$$

Таким образом, управление (2.4), (2.7) решает задачу программного перевода системы из состояния x_0 в 0 за время T.

Замкнем систему (2.1) управлением (2.4), (2.7) и проинтегрируем ее в пределах от некоторого текущего момента t до T. Замкнутая система имеет вид

$$\dot{x} = P(t)x + Q(t)(B^T(t)c + v(t)) + f(t). \tag{2.8}$$

При этом момент t становится начальным моментом времени, а соответствующее значение фазового вектора $x(t)$ — вектором начальных данных. Обозначим через $\widetilde{t}$ формальную текущую переменную интегрирования. Тогда

решение задачи Коши

$$\widetilde{t}_0 = t, \quad x(\widetilde{t}_0) = x(t),$$

системы (2.8) имеет вид

$$x(\widetilde{t}, t, x(t)) = Y(\widetilde{t})\Bigg(Y^{-1}(t)x(t)+ \\ + \int_t^{\widetilde{t}} Y^{-1}(\tau)\left(Q(\tau)(B^T(\tau)c + v(\tau)) + f(\tau)\right)d\tau\Bigg). \tag{2.9}$$

Поскольку (2.9) — это представление программного движения системы (2.1), (2.4), (2.7) на отрезке $t \in [t, T]$, то $x(t, t, x(t)) = x(t)$, $x(T, t, x(t)) = 0$ (см. рис. 5).

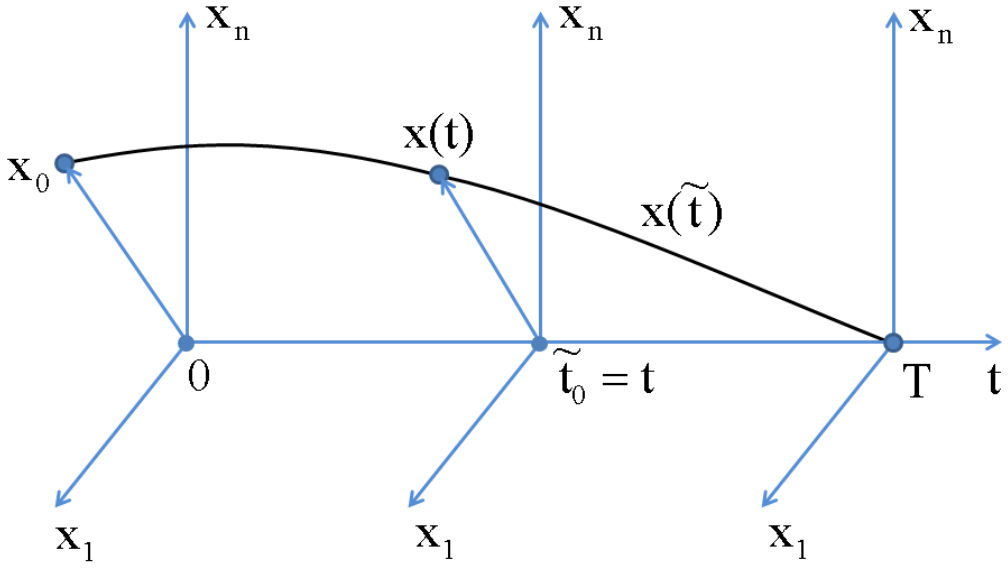

Рис. 5. Программное движение системы (2.1), замкнутой управлением (2.4), (2.7)

При $\widetilde{t} = T$ из (2.9) получаем

$$0 = Y(T)\Bigg(Y^{-1}(t)x(t) + \int_t^T Y^{-1}(\tau)\left(Q(\tau)(B^T(\tau)c + v(\tau)) + f(\tau)\right)d\tau\Bigg). \tag{2.10}$$

Из уравнения (2.10) выразим вектор

$$c = -A^{-1}(t, T)\left(Y^{-1}(t)x(t) + \xi(t)\right), \tag{2.11}$$

где

$$\xi(t) = \int_t^T B(\tau)v(\tau)d\tau + \int_t^T Y^{-1}(\tau)f(\tau)d\tau.$$

Подставим теперь (2.11) в (2.4), получим искомое представление для вектора управления

$$u(t,x) = B^T(t)\left(-A^{-1}(t,T)\left(Y^{-1}(t)x(t) + \xi(t)\right)\right) + v(t)$$

или

$$u(t,x) = M(t)x(t) + N(t), \tag{2.12}$$

где

$$M(t) = -B^T(t)A^{-1}(t,T)Y^{-1}(t), \quad N(t) = v(t) - B^T(t)A^{-1}(t,T)\xi(t). \tag{2.13}$$

Замкнем систему (2.1) позиционным управлением (2.12)

$$\dot{x} = \big(P(t) + Q(t)M(t)\big)x + Q(t)N(t) + f(t), \tag{2.14}$$

Для построения общего решения системы (2.14) не обязательно ее интегрировать, достаточно приравнять правые части формул (2.7) и (2.11), а затем выразить вектор $x(t)$:

$$x(t) = Y(t)\left(A(t,T)A^{-1}(0,T)\left(x_0 + \int_0^T Y^{-1}(\tau)f(\tau)d\tau\right) - \xi(t)\right). \tag{2.15}$$

Убедиться в том, что (2.15) — это общее решение системы (2.14) в форме Коши можно простой подстановкой. Кроме того, очевидно, что $x(0) = x_0$ и $x(T) = 0$.

Таким образом, доказано следующее утверждение.

Теорема 2 [3, с. 220]. *Если столбцы матрицы $B^T(t)$ — линейно независимые векторные функции в любом промежутке $[t,T]$, $t \geqslant 0$, то существует семейство управлений (2.12), зависящее от произвольной векторной функции $v(t)$, удовлетворяющей условию (2.5), обладающее тем свойством, что любое решение системы (2.1) $x = x(t)$ при управлении (2.12) будет обладать свойством $x = x_0$ при $t = 0$ и $x = 0$ при $t = T$, где x_0 — произвольное начальное фазовое состояние.*

§ 2.2. Алгоритм построения позиционного управления

Доказательство теоремы 2, проведенное выше, дает конструктивный метод построения управления (2.12). Для этого нужно:

1. Построить фундаментальную матрицу $Y(t)$ однородной системы $\dot{x} = P(t)x$ и найти обратную к ней $Y^{-1}(t)$. С этой целью можно использовать сопряженную систему.

2. Последовательно построить матрицы $B(t)$, $A(0,T)$, $A(t,T)$.

3. Найти некоторое решение интегрального уравнения (2.5). Это можно сделать из различных соображений, в том числе и методом неопределенных коэффициентов, раскладывая элементы векторной функции $v(t)$ в виде линейных комбинаций заданного набора известных функций. При этом всегда можно положить $v(t) \equiv 0$, так как уравнение (2.5) имеет нулевое решение.

4. Построить векторную функцию $\xi(t)$ (см. (2.11)).

5. По формулам (2.13) вычислить матрицы $M(t)$ и $N(t)$. Результат подставить в представление (2.12).

§ 2.3. Линейная однородная стационарная система

Рассмотрим важный частный случай, когда система (2.1) имеет вид

$$\dot{x} = Px + Qu, \tag{2.16}$$

где P, Q — постоянные матрицы соответствующих размеров, $f(t) = 0$.

Известно, что функционал

$$\int_0^T u^T(\tau)u(\tau)d\tau, \tag{2.17}$$

заданный на семействе управлений (2.4), принимает минимальное значение при $v(t) \equiv 0$, поэтому будем строить позиционное управление при $v(t) \equiv 0$. Формулы (2.12), (2.13) в этом случае имеют вид

$$u(t,x) = M(t)x(t), \tag{2.18}$$

$$M(t) = -B^T(t)A^{-1}(t,T)Y^{-1}(t). \tag{2.19}$$

Запишем замкнутую систему (2.16), (2.18)

$$\dot{x} = \big(P - QB^T(t)A^{-1}(t,T)Y^{-1}(t)\big)x, \tag{2.20}$$

Решение системы (2.20) с учетом (2.15) имеет вид

$$x(t,0,x_0) = Y(t)A(t,T)A^{-1}(0,T)x_0. \tag{2.21}$$

Поскольку функция (2.21) обладает свойством $x(0,0,x_0) = x_0$, $x(T,0,x_0) = 0$, то управление (2.18), (2.19) является искомым позиционным управлением для системы (2.16).

Описанный выше алгоритм в данном случае упрощается, в нем не нужно выполнять пункты 3, 4.

Пример. Рассмотрим простейший вариант системы (2.16). Пусть

$$\dot{x} = 2x + 6u.$$

Здесь $P = 2$, $Q = 6$. Зададим граничные условия: $T = 10$, $x(0) = x_0$, $x(10) = 0$.

Реализуя алгоритм, последовательно найдем

$$Y(t) = e^{2t}, \quad Y^{-1}(t) = e^{-2t}, \quad B(t) = 6e^{-2t},$$

$$A(t,T) = 9\left(e^{-4t} - e^{-40}\right), \quad A^{-1}(t,T) = \frac{1}{9\left(e^{-4t} - e^{-40}\right)}, \quad A^{-1}(0,T) = \frac{1}{9\left(1 - e^{-40}\right)}.$$

В результате позиционное управление (2.18), (2.19) имеет вид

$$u(t,x) = -\frac{2e^{-4t}}{3\left(e^{-4t} - e^{-40}\right)}x.$$

Замкнутая система (2.20), соответственно,

$$\dot{x} = \left(2 - \frac{4e^{-4t}}{e^{-4t} - e^{-40}}\right)x,$$

Ее решение согласно (2.21) имеет вид

$$x(t,0,x_0) = e^{2t}\left(\frac{e^{-4t} - e^{-40}}{1 - e^{-40}}\right)x_0.$$

Это решение обладает свойством $x(0) = x_0$, $x(10) = 0$.

§ 2.4. Численный анализ свойств позиционного управления

Пример, приведенный в §2.3, наглядно показывает основное предельное свойство построенного позиционного управления, точнее его коэффициента усиления:

$$\lim_{t\to 10} \frac{2e^{-4t}}{3\left(e^{-4t} - e^{-40}\right)} = +\infty.$$

Для скалярного уравнения это очевидно, так как матрица

$$M(t) = -B^T(t)A^{-1}(t,T)Y^{-1}(t)$$

(см. (2.19)). При этом функции $B^T(t)$ и $Y^{-1}(t)$ ограничены при $t \to 10$, а

$$\lim_{t\to 10} A^{-1}(t,T) = \lim_{t\to 10} \frac{1}{9\left(e^{-4t} - e^{-40}\right)} = +\infty.$$

Показать аналитически свойство матрицы $M(t)$

$$\lim_{t\to T} \|M(t)\| = +\infty.$$

пока не удалось, поскольку в общем виде нет явного представления элементов этой матрицы. Поэтому дальнейшие усилия направлены на реализацию численного алгоритма построения и анализа предельных свойств матрицы $M(t)$.

В пакете MATLAB разработана программа (см. Приложение 2), которая позволяет вычислять матрицу позиционного управления $M(t)$, строить график ее нормы как функции времени и выполняет последующие шаги алгоритма §2.2.

Покажем работу программы для следующего модельного примера. Пусть линейная управляемая система имеет вид

$$\dot{x} = \begin{pmatrix} 1 & 2 \\ 0 & -1 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u, \quad x_0 = \begin{pmatrix} 5 \\ 6 \end{pmatrix}, \quad x(T) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad T = 10.$$

На рис. 6 представлен график нормы матрицы $M(t)$ позиционного управления (2.18), (2.19), построенного для данного примера.

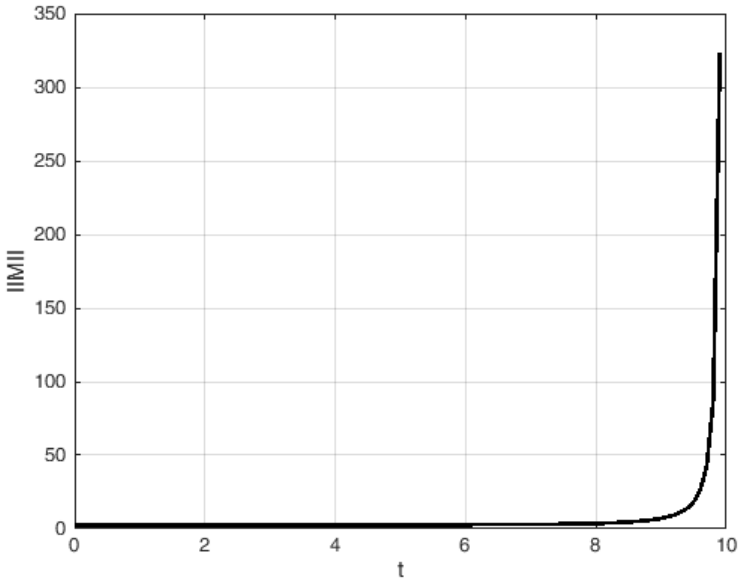

Рис. 6. Норма матрицы позиционного управления

На рис. 7 показаны интегральные кривые линейной системы из данного примера, замкнутой позиционным управлением. На рис. 8 графики совмещены, чтобы наглядно показать интересный нюанс. А именно, норма матрицы $M(t)$ начинает существенно расти лишь на небольшом последнем участке процесса управления.

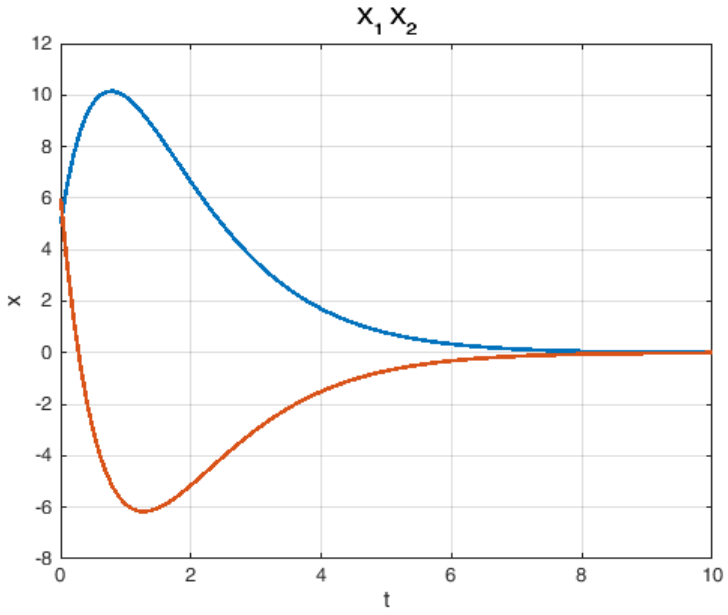

Рис. 7. Интегральные кривые линейной системы, замкнутой позиционным управлением

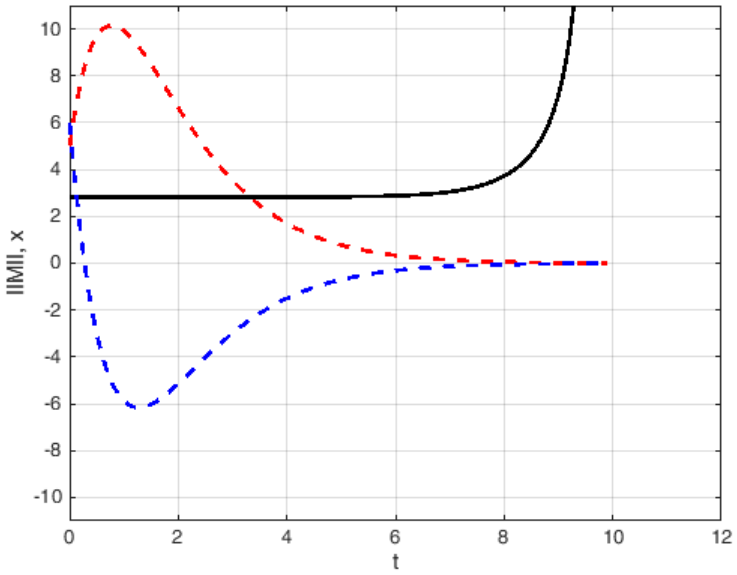

Рис. 8. Норма позиционного управления и интегральные кривые замкнутой системы

Далее можно сформулировать две взаимосвязанные задачи.

Задача 1. Допустим, норма матрицы $M(t)$ ограничена наперед заданной константой α, т. е. $\|M(t)\| \leqslant \alpha$. Возникает вопрос: в какую окрестность начала координат можно перевести систему, используя управления, удовлетворяющие данному ограничению, при фиксированном $x(0) = x_0$?

Задача 2. Допустим, задана ε-окрестность начала координат. Требуется выяснить минимально достаточное значение нормы матрицы $M(t)$, при котором позиционное управление обеспечит перевод замкнутой системы в эту ε-окрестность.

Для решения поставленных задач необходимо следить за нормой матрицы $M(t)$ на каждом шаге алгоритма и одновременно вычислять норму вектора фазового состояния в соответствующий момент времени. Разработанная программа позволяет это делать. На рис. 9 показана зависимость нормы вектора текущего состояния системы от нормы матрицы коэффициентов усиления позиционного управления.

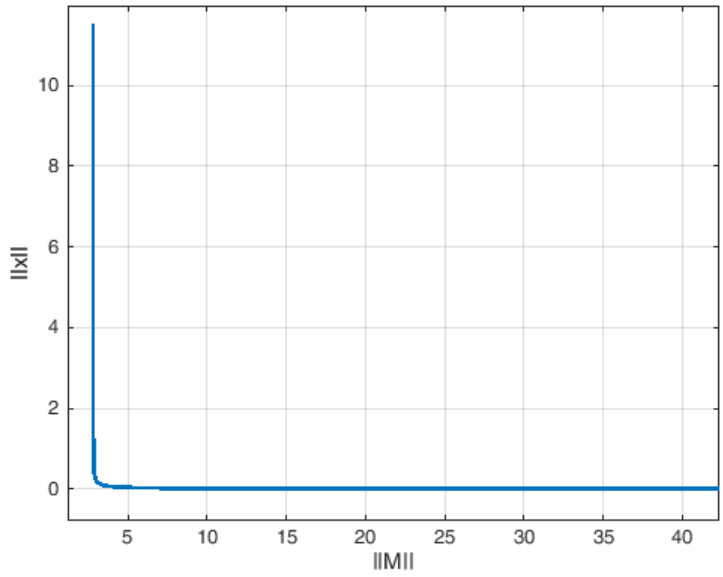

Рис. 9. Зависимость нормы вектора текущего состояния от нормы матрицы позиционного управления

Таким образом, для решения задачи 1 необходимо следить за нормой матрицы $M(t)$, по заданной величине α засечь момент времени t_α, когда достигается равенство $\|M(t_\alpha)\| = \alpha$. Тогда систему можно перевести в ε-окрестность начала координат, где $\varepsilon = \|x(t_\alpha)\|$. Задача 2 решается аналогично, только необходимо отслеживать норму текущего состояния системы

и определять момент времени t_ε, когда она становится равна ε. Затем найти величину $\alpha = \|M(t_\varepsilon)\|$.

§ 2.5. Комбинированный алгоритм синтеза управлений в линейных системах

Под комбинированным алгоритмом будем понимать алгоритм, в котором на разных этапах процесса управления используются разные типы управляющего воздействия на объект управления. В данном случае речь идет о переводе линейной системы (2.16) в начало системы координат из некоторого начального состояния $x = x_0$ за время T.

Потребность в организации комбинированного алгоритма возникает по двум причинам. Во-первых, требуется построить позиционное управление, так как в дальнейшем оно будет использоваться для синтеза управлений в классе билинейных систем. А во-вторых, необходимо учитывать предельные свойства этого управления и реальные ограничения на матрицу коэффициентов усиления, которые были описаны в § 2.4.

Последовательность шагов комбинированного алгоритма представляет собой некоторое расширение алгоритма построения позиционного управления из § 2.2. А именно:

Этап 1. Выполняются шаги алгоритма построения позиционного управления из §2.2 до тех пор, пока выполняется условие $\|M(t)\| \leqslant \alpha$ при заданном значении константы α.

Этап 2. Фиксируется момент времени t_α, когда достигается равенство $\|M(t_\alpha)\| = \alpha$ и определяется соответствующее состояние системы $x(t_\alpha)$.

Этап 3. На данном этапе точка $x(t_\alpha)$ считается начальной. Для системы

(2.16) ставится задача программного перевода из состояния $x_0 = x(t_\alpha)$ при $t = t_\alpha$ в состояние $x = 0$ при $t = T$. Для решения этой задачи используются формулы (2.4), (2.7).

Можно считать, что на этапе 2 данного алгоритма определяется момент перехвата управления, когда позиционное управление исчерпало свой допустимый ресурс. Далее включается стандартный блок реализации обычного программного управления.

Моделирующая программа реализует предложенный алгоритм. Для примера из § 2.4 было добавлено ограничение на позиционное управление $\alpha = 10$. Соответственно был найден момент перехвата управления $t_\alpha = 9{,}3$. Далее, на отрезке $t \in [9{,}3;\ 10]$ включилось программное управление. Результаты численного моделирования приведены на рис. 10. Красным цветом обозначено позиционное управление. При этом сплошная линия — координата x_1 фазового вектора, пунктир — координата x_2 фазового вектора, звездочка — момент перехвата управления, синим цветом обозначен этап программного управления.

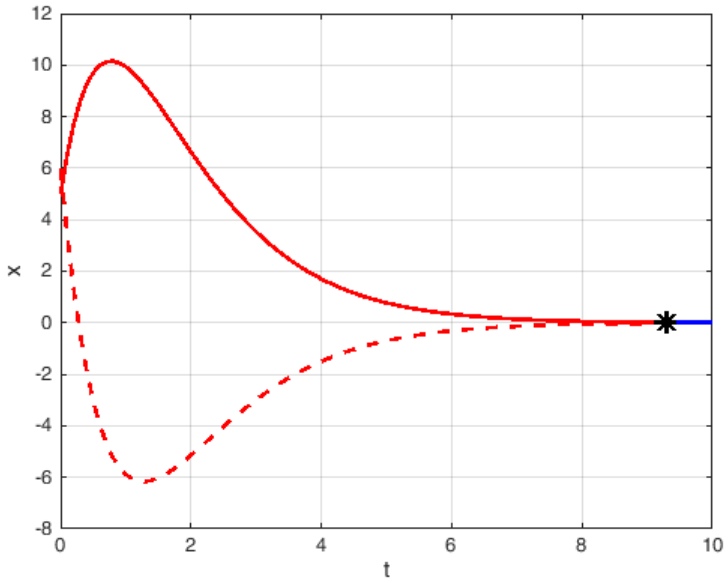

Рис. 10. Реализация комбинированного алгоритма управления

§ 2.6. Выводы по главе 2

Отметим основное свойство позиционного управления (2.18), (2.19). Линейная система, замкнутая этим управлением, при любом начальном векторе $x(0) = x_0$ (в том числе и возмущенном по отношению к некоторому фиксированному значению) имеет решение (2.21), проходящее через точку $x(T) = 0$. Это означает, с одной стороны, что нет необходимости дополнительно решать задачу стабилизации. С другой стороны, норма матрицы коэффициентов усиления $M(t)$ позиционного управления неограниченно возрастает по мере приближения к финальной точке. Это так, поскольку при такой конструкции управления точка $x = 0$ является особой точкой замкнутой системы, так как через нее проходит бесконечно много интегральных кривых.

Для преодоления этого негативного свойства позиционного управления и обеспечения его конструктивного применения в прикладных задачах, возникла идея комбинированного алгоритма, когда на первом этапе, пока норма матрицы $M(t)$ не превзошла допустимых значений, применяется позиционное управление. Оно гарантированно (без дополнительной стабилизации) переводит систему в достаточно близкую окрестность начала координат, а затем включается стандартное программное управление, которое переводит замкнутую систему в ноль и, таким образом, завершает решение поставленной задачи. Как показали численные эксперименты, длина финального участка невелика (см. рис. 10), т. е. программное движение также не требует дополнительной стабилизации, хотя система стабилизации и может присутствовать на случай «форс-мажорных» обстоятельств.

Комбинированный алгоритм важен не только для линейных систем. Далее покажем, как позиционное управление применять в билинейных системах.

Глава 3. Синтез управлений в билинейных системах

В этом разделе показано применение линейных вспомогательных моделей для решения задачи синтеза в билинейных однородных системах. Приведен численный пример реализации алгоритма.

§ 3.1. Билинейная однородная система общего вида

Рассмотрим билинейную однородную управляемую систему

$$\dot{x} = P(t)x + Q(t)U(t)x, \tag{3.1}$$

где x — вектор фазового состояния размерности n; матрицы $P(t)$, $Q(t)$ размерностей $(n \times n)$, $(n \times r)$ имеют вещественные, непрерывные и ограниченные при $t \geqslant 0$ элементы; $U(t)$ — $(r \times n)$-матрица управлений.

Используя матрицы ее коэффициентов, запишем вспомогательную линейную систему

$$\dot{x} = P(t)x + Q(t)u, \tag{3.2}$$

где u — r-мерный вектор управления.

Поскольку система (3.2) однородная, то для решения задачи синтеза позиционного управления, переводящего ее из заданного фазового состояния $x = x_0$ при $t = 0$ в конечное состояние $x = 0$ при $t = T$, можно использовать упрощенный алгоритм §2.3. Предположим, что управление (2.18), (2.19) существует и построено. Тогда замкнем им систему (3.2):

$$\dot{x} = P(t)x + Q(t)M(t)x. \tag{3.3}$$

Сравнивая системы (3.1) и (3.3), получаем матричное уравнение

$$U(t) = M(t). \tag{3.4}$$

Утверждение 1 [33]. *Если столбцы матрицы $B^T(t)$ — линейно независимые векторные функции в любом промежутке $[t, T]$, $t \geqslant 0$, а матричное уравнение (3.4) имеет хотя бы единственное решение $\overline{U}(t)$, то при управлении $U = \overline{U}(t)$ в системе (3.1) существует решение, обладающее свойством $x = x_0$ при $t = 0$ и $x = 0$ при $t = T$.*

Заметим, что утверждение 1 по сути является следствием теоремы 2. Простота идеи построения уравнения (3.4) не отменяет трудности применения теоремы 2 в задачах синтеза управлений в различных классах билинейных систем. Действительно, в простейшем случае, когда все элементы матрицы $U(t)$ управляемые параметры, мы получим решение в виде матрицы $M(t)$. Тем не менее, если хотя бы один из элементов матрицы $U(t)$ равен нулю в качестве структурного элемента системы управления, то нельзя формально положить $U(t) = M(t)$.

§ 3.2. Билинейная система с линейной структурой параметров

Рассмотрим важный частный случай. Будем предполагать, что в билинейной системе (3.1) каждый элемент u_{ij}, $i = \overline{1, r}$, $j = \overline{1, n}$, матрицы $U(t)$ является линейной формой заданного набора параметров $v_1, \ldots, v_m$. Это предположение можно рассматривать как ограничение на структуру матрицы $U(t)$, которая часто встречается в приложениях. Таким образом, мы имеем m параметров для решения задачи управления и соответствующие представления

$$u_{ij} = \sum_{k=1}^{m} \alpha_{ij,k} v_k, \tag{3.5}$$

где $\alpha_{ij,k}$ — константы при $i = \overline{1, r}$, $j = \overline{1, n}$, $k = \overline{1, m}$. В этом случае матричное

уравнение (3.4) принимает вид

$$\sum_{k=1}^{m} \alpha_{ij,k} v_k = \mu_{ij}(t), \quad i = \overline{1,r},\ j = \overline{1,n}. \tag{3.6}$$

Систему (3.6) можно записать в стандартной матричной форме

$$\overline{A} v = \overline{\mu}, \tag{3.7}$$

где $v = (v_1, \ldots, v_m)^T$ — вектор неизвестных параметров, $\overline{A} = \{\alpha_{ij,k}\}$ — $(rn \times m)$-матрица, $\overline{\mu}$ — rn-вектор. Для системы (3.7) существует ранговый критерий совместности. Таким образом, справедливо

Утверждение 2. *Если выполнены следующие условия: 1) столбцы матрицы $B^T(t)$ — линейно независимые векторные функции в любом промежутке $[t, T]$, $t \geqslant 0$; 2)* $\operatorname{rank} \overline{A} = \operatorname{rank} (\overline{A}, \overline{\mu})$. *Тогда при управлении (3.5), где $v_1, \ldots, v_m$ — решения системы (3.7), в системе (3.1) существует решение, обладающее свойством $x = x_0$ при $t = 0$ и $x = 0$ при $t = T$.*

§ 3.3. Численная реализация

В качестве тестового примера рассмотрим задачу синтеза для билинейной системы

$$\dot{x} = \begin{pmatrix} 1 & 2 \\ 0 & -1 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} u_1 & u_2 \end{pmatrix} x.$$

Вспомогательная линейная система (3.2) имеет вид

$$\dot{x} = \begin{pmatrix} 1 & 2 \\ 0 & -1 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u.$$

Следуя алгоритму §2.2, последовательно найдем

$$Y(t) = \begin{pmatrix} e^t & e^{-t}\left(e^{2t}-1\right) \\ 0 & e^{-t} \end{pmatrix},$$

$$Y^{-1}(t) = \begin{pmatrix} e^{-t} & -e^{-t}\left(e^{2t}-1\right) \\ 0 & e^{t} \end{pmatrix}.$$

Матрицы $B(t)$, $A(t,T)$, $A^{-1}(t,T)$, $M(t)$ были найдены численно в пакете MATLAB. Интегральные кривые замкнутой системы показаны на рис. 11.

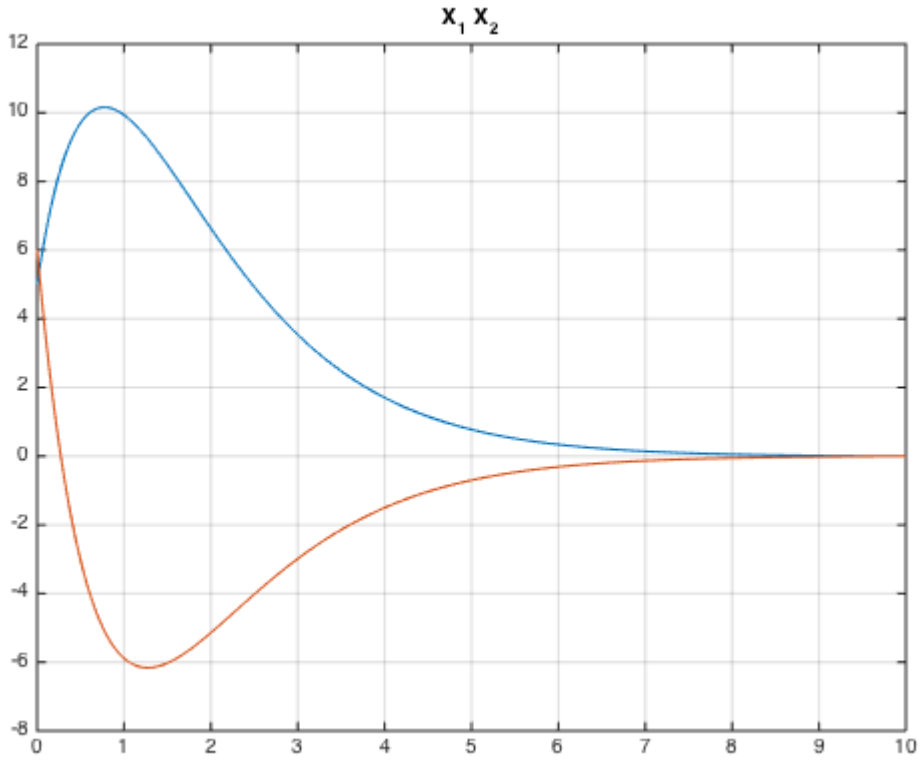

Рис. 11. Интегральные кривые замкнутой билинейной системы

Они соответствуют начальным данным $x_1(0) = 5$, $x_2(0) = 6$ и $T = 10$.

Заключение

Перечислим основные результаты работы, выносимые на защиту:

- проведен сравнительный анализ основных методов синтеза управлений в линейных системах, отмечены их сильные и слабые стороны в плане практической реализации и области приложений;
- проведен численный анализ предельных свойств позиционного управления в линейных системах, построенного по методу Зубова;
- разработан комбинированный алгоритм синтеза управлений в линейных системах;
- предложен подход к решению задач синтеза управлений в билинейных системах;
- в пакете MATLAB разработан набор программ, реализующих алгоритмы описанных методов;
- работа программ протестирована на модельных примерах;
- результаты работы частично опубликованы.

Литература

1. Понтрягин Л. С., Болтянский В. Г., Гамкрелидзе Р. В., Мищенко Е. Ф. Математическая теория оптимальных процессов. М.: Наука, 1969. 384 с.
2. Красовский Н. Н. Теория управления движением. М.: Наука, 1968. 476 с.
3. Зубов В. И. Лекции по теории управления. СПб.: Лань, 2009. 496 с.
4. Зубов В. И. Математические методы исследования систем автоматического регулирования. Л.: Машиностроение, 1974. 336 с.
5. Калман Р. Е. Об общей теории систем управления. Труды I Междунар. конгресса ИФАК. М.: Изд-во АН СССР, 1961. Т. 2. С. 521–547.
6. Калман Р., Фалб П., Арбиб М. Очерки по математической теории систем. М.: Мир, 1971. 400 с.
7. Андреев Ю. Н. Управление конечномерными линейными объектами. М.: Наука, 1976. 424 с.
8. Смирнов Е. Я. Некоторые задачи математической теории управления. Л.: Изд-во Ленингр. ун-та, 1981. 200 с.
9. Габасов Р. Ф., Кириллова Ф. М. Качественная теория оптимальных процессов. М.: Наука, 1971. 508 с.
10. Габасов Р., Кириллова Ф. М. К проблеме синтеза оптимальных систем // Известия ВУЗ. Математика. 2001. № 12. С. 10–20.
11. Альсевич В. В., Габасов Р., Глушенков В. С. Оптимизация линейных экономических моделей. Минск: Изд-во БГУ, 2000. 211 с.

12. Габасов Р., Кириллова Ф. М., Ружицкая Е. А. Демпфирование и стабилизация маятника при больших начальных возмущениях // Известия РАН. Теория и системы управления. 2001. № 6. С. 29–38.

13. Габасов Р., Ружицкая Е. А. Стабилизация динамических систем с обеспечением дополнительных свойств переходных процессов // Кибернетика и системный анализ. 2001. № 3. С. 139–151.

14. Балашевич Н. В., Габасов Р., Кириллова Ф. М. Численные методы программной и позиционной оптимизации линейных систем управления // Журн. вычисл. математики и мат. физики. 2000. Т. 40. № 6. С. 838–859.

15. Попков А. С., Баранов О. В. Об оптимальном управлении вращательным движением вала электродвигателя // Процессы управления и устойчивость. 2014. Т. 1. № 1. С. 31–36.

16. Popkov A. S., Baranov O. V., Smirnov N. V. Application of adaptive method of linear programming for technical objects control // 2014 International Conference on Computer Technologies in Physical and Engineering Applications (ICCTPEA), 2014. P. 141–142.

17. Баранов О. В., Попков А. С., Смирнов Н. В. Оптимальная стабилизация квадрокоптера в режиме реального времени // Устойчивость и процессы управления: Материалы III международной конференции, посвященной 85-летию со дня рождения чл.-корр. РАН В.И. Зубова. СПб, 2015. С. 115–116.

18. Клюенков А. Л. Реализация адаптивного метода в одной задаче оптимального управления // Процессы управления и устойчивость. 2015. Т. 2. № 1. С. 53–58.

19. Mohler R. R. Natural bilinear control process // IEEE Trans. Automatic Control. 1970. Vol. 15, No. 3. P. 192–197.

20. Mohler R. R. Bilinear Control Process. New York, USA: Academic Press, 1973.

21. Mohler R. R., Ruberti A. Recent Developments in Variable Structure Systems, Economics, and Biology. New York, USA: Springer, 1978.

22. Krener A. J. Bilinear and nonlinear realizations of input-output maps // SIAM J. Control. 1975. Vol. 13, No. 4. P. 827–834.

23. Svoronos S., Stephanopoulos G., Aris R. Bilinear approximation of general non-linear dynamic systems with linear inputs // Int. J. Control. 1980. Vol. 31, No. 1. P. 109–126.

24. Peresada V. P., Smirnov N. V., Smirnova T. E. Development control of a multicommodity economy based on the dynamical input-output model // Вестник Санкт-Петербургского университета. Серия 10: Прикладная математика. Информатика. Процессы управления. 2014. № 4. С. 119–132.

25. Петросян Л. А., Захаров В. В. Математические модели в экологии. СПб.: Изд-во С.-Петербург. ун-та, 1997. 253 с.

26. Зубов В. И. Синтез многопрограммных устойчивых управлений // Докл. АН СССР. 1991. Т. 318. № 2. С. 274–277.

27. Smirnov N. V. Multiprogram control for dynamic systems: A point of view // ACM International Conference Proceeding Series, Joint International Conference on Human-Centered Computer Environments, HCCE 2012, 2012. P. 106–113.

28. Ляпунов А. М. Общая задача об устойчивости движения. М. – Л.: ОНТИ, 1935. 386 с.

29. Демидович Б. П. Лекции по математической теории устойчивости. СПб.: Лань, 2008. 480 с.

30. Александров А. Ю., Александрова Е. Б., Екимов А. В., Смирнов Н. В. Сборник задач и упражнений по теории устойчивости. Учеб. пособие, 3-е издание. СПб.: Лань, 2016. 160 с.

31. Смирнов Н. В., Смирнова Т. Е., Тамасян Г. Ш. Стабилизация программных движений при полной и неполной обратной связи. Учеб. пособие. СПб.: Лань, 2016. 128 с.

32. Жабко А. П., Котина Е. Д., Чижова О. Н. Дифференциальные уравнения и устойчивость. Учеб. пособие. СПб.: Лань, 2015. 320 с.

33. Смирнов А. Н., Смирнов Н. В., Смирнова Т. Е. Синтез управлений в билинейной системе на основе вспомогательной линейной модели // Устойчивость и колебания нелинейных систем управления: Материалы XIII Международной конференции (1–3 июня 2016 г., Москва) / ред. В. Н. Тхай. М.: ИПУ РАН, 2016. С. 333–335.

Приложения

Приложение 1. Реализация алгоритма построения программного и стабилизирующего управления

Входные данные: 1) матрица P и вектор Q коэффициентов системы (1.1); 2) векторная функция возмущений $f(t)$; 3) граничные условия из постановки задачи t_0, T, $x(t_0) = x_0$, $x(t_0 + T) = x_1$.

Выходные данные: 1) представление программного управления $u(t)$; 2) графики интегральных кривых системы, замкнутой построенным программным управлением; 3) представление стабилизирующего управления: 4) графики интегральных кривых системы, замкнутой комбинированным управлением, с учетом работы блока стабилизации.

```
% Построение программного управления
clear all
close all
clc
syms x(t) y(t)
% Входные данные
t0=0;
t1=20;
T=t1;
xStart = [1; 1];
xEnd   = [3; 4];
Q = [0; 1];
P = [-5  -5; 3  -1];
global f
f = 0.1*[sin(t); cos(t)];
% Построение фундаментальной матрицы
x0=1;
y0=0;
z1 = dsolve(...
                  diff(x) == -5*x-5*y,...
                  diff(y) == 3*x-y,...
                  x(t0)==x0,...
                  y(0)==y0... );

x02=0;
y02=1;
```

```
z2 = dsolve(...
                diff(x) == -5*x-5*y,...
                diff(y) == 3*x-y,...
                x(t0)==x02,...
                y(0)==y02... );

disp('Y = ')
fundMatrY = [z1.x z2.x;...
             z1.y z2.y];
fundObratka = inv(fundMatrY);

% Интегральное уравнение
B = fundObratka * Q;
BforInt = B*B';

B11 = matlabFunction(BforInt(1,1));
B12 = matlabFunction(BforInt(1,2));
B21 = matlabFunction(BforInt(2,1));
B22 = matlabFunction(BforInt(2,2));

A11 = int(B11,t,t0,t0+T);
A12 = int(B12,t,t0,t0+T);
A21 = int(B21,t,t0,t0+T);
A22 = int(B22,t,t0,t0+T);

% Формирование и решение системы линейных алгебраических уравнений (1.4)
A = [A11 A12;A21 A22];

fundObratkaF = fundObratka*f;
fundObratkaF11 = matlabFunction(fundObratkaF(1,1));
fundObratkaF21 = matlabFunction(fundObratkaF(2,1));
IntVozm11 = int(fundObratkaF11, t, t0,t0+T);
IntVozm21 = int(fundObratkaF11, t, t0,t0+T);
IntVozm = [IntVozm11; IntVozm21];

b = subs(fundObratka,t,t0+T)*xEnd - xStart - IntVozm;
bout = eval(b);
Aout = eval(A);

c = linsolve(A,b);
cout = eval(c);
% Программное управление и программное движение
global up1
up = eval(B'*c);
up1 = matlabFunction(up);

forIntXp = fundObratka*(Q*up + f);
forIntXp11 = matlabFunction(forIntXp(1,1));
```

```
75  forIntXp21 = matlabFunction(forIntXp(2,1));
76  IntXp11 = int(forIntXp11, t, t0,t);
77  IntXp21 = int(forIntXp21, t, t0,t);
78  IntXp = [IntXp11;...
79           IntXp21];
80
81  global xp
82  xp = fundMatrY * (xStart + IntXp);
83
84  % Построение графиков программных движений
85  ttt = t0:0.1:t0+T;
86  figure(1)
87  xxx1 = subs(xp(1),t, ttt);
88  plot(ttt,eval(xxx1),'LineWidth',2)
89  grid on
90  title('x_1')
91  xlabel('t')
92  ylabel('x_1')
93  grid on
94  print('x1 prog','-dpng')
95
96  figure(2)
97  xxx2 = subs(xp(2),t, ttt);
98  plot(ttt,eval(xxx2),'LineWidth',2)
99  grid on
100 title('x_2')
101 xlabel('t')
102 ylabel('x_2')
103 grid on
104 print('x2 prog','-dpng')
105
106 % Построение стабилизирующего управления
107 % Матрицы коэффициентов системы
108 A1 = P;
109 B1 = Q;
110 C1 = zeros(2);
111 D1 = 0;
112 sys = ss(A1,B1,C1,D1);
113 Q1 = [0.2  0;...
114         0  0.2];
115 R1 = 0.1;
116 N1 = zeros(2,1);
117 % Вычисление матрицы коэффициентов усиления обратной связи (1.6)
118 global C_upr;
119 C_upr = -lqr(sys,Q1,R1,N1)
120
121 % Построение графиков интегральных кривых замкнутой системы
122 [T2,Y2] = ode45(@syst2,[0  20],[1  1]);
```

```
figure(3)
plot(T2,Y2(:,1),'-','LineWidth',2)
hold on
plot(T2,Y2(:,2),'-.','LineWidth',2)
grid on

function dy = syst2(t,y)
global C_upr;
dy = zeros(2,1);
dy = (P+Q*C_upr)*y+Q*up-Q*C_upr*xp+f;
end
```

Приложение 2. Реализация алгоритма построения позиционного управления и анализ его свойств

Входные данные: 1) матрица P и вектор Q коэффициентов системы (2.1); 2) граничные условия постановки задачи t_0, T, $x(t_0) = x_0$, $x(t_0+T) = 0$.

Выходные данные: 1) матрица $M(t)$ коэффициентов позиционного управления; 2) интегральные кривые линейной системы, замкнутой позиционным управлением; 3) график зависимости нормы решения от нормы матрицы $M(t)$; 4) момент времени перехвата управления в комбинированном алгоритме; 5) интегральные кривые билинейной системы (3.1), замкнутой управлением $U(t) = M(t)$.

```
% Построение позиционного управления
clear all
close all
clc
syms x(t) y(t)

% Входные данные
t0=0;
t1=10;
T=t1;
xStart = [5; 6];
xEnd   = [0; 0];
Q = [0; 1];
P = [1   2; 0   -1];
```

```
% Построение фундаментальной матрицы
x0=1;
y0=0;
z1 = dsolve(...
                    diff(x) == x+2*y,...
                    diff(y) == -y,...
                    x(t0)==x0,...
                    y(0)==y0...  );
x02=0;
y02=1;
z2 = dsolve(...
                    diff(x) == x+2*y,...
                    diff(y) == -y,...
                    x(t0)==x0,...
                    y(0)==y0...  );
disp('Y = ')
fundMatrY = [z1.x  z2.x;...
               z1.y  z2.y];
fundObratka = inv(fundMatrY);

% Построение матриц B(t), A(t,T)
B=fundObratka*Q
BforInt=B*B'
B11 = matlabFunction(BforInt(1,1))
B12 = matlabFunction(BforInt(1,2))
B21 = matlabFunction(BforInt(2,1))
B22 = matlabFunction(BforInt(2,2))
A11 = int(B11,t,t,T)
A12 = int(B12,t,t,T)
A21 = int(B21,t,t,T)
A22 = int(B22,t,t,T)
A = [A11 A12;A21 A22]
Ainv = inv(A);
Ainv_t0 = subs(Ainv,t,t0);

% Построение матрицы M(t) коэффициентов позиционного управления, рис. 6
Umatr = -B'*Ainv*fundObratka;

% Решение (2.21) системы (2.16), замкнутой позиционным управлением (2.18), (2.19), рис. 7
res = fundMatrY * A * Ainv_t0 * xStart;

% Матрица билинейной системы (3.1), замкнутой управлением U(t)=M(t), интегральные кривые на рис. 11
Pclose = P+Q*Umatr;

% Анализ предельных свойств позиционного управления
% Поиск момента времени, когда норма матрицы M становится больше заданного числа alpha
alpha = 10;
for ti = 0:0.1:T
```

```
63      u_subs_norm = subs(norm(Umatr,2),t,ti);
64      if subs(norm(Umatr,2),t,ti) < alpha
65  figure(8)
66          plot(ti,u_subs_norm,'k*')
67  grid on
68  hold on
69      else
70          ti
71          break
72      end
73  end
74
75  % Построение графика зависимости нормы решения от нормы матрицы M(t), рис. 9
76  t01 = 0.000001:0.1:T;
77  res_t01_norm = subs(norm(res,2),t,t01);
78  Umatr_t01_norm = subs(norm(Umatr,2),t,t01);
79  plot(Umatr_t01_norm, res_t01_norm,'LineWidth',2)
80  grid on
81  xlabel('||M||')
82  ylabel('||x||')
83
84  % Перехват управления. Реализация комбинированного алгоритма управления, рис. 10
85  t11 = 9.3:0.1:T;
86  t22 = 0:0.1:9.3;
87  u_prog_subs1 = subs(u_prog2,t,t11);
88  x_prog1 = subs(z3.x,t,t11);
89  x_prog2 = subs(z3.y,t,t11);
90  plot(t11,x_prog1,'b--','LineWidth',2)
91  plot(t11,x_prog2,'b','LineWidth',2)
92  hold on
93  grid on
94
95  plot(t22,(subs(res(1),t,t22)),'r','LineWidth',2)
96  plot(t22,(subs(res(2),t,t22)),'r--','LineWidth',2)
97  hold on
98  plot(ti,subs(z3.x,t,ti),'*k','MarkerSize',10,'LineWidth',2)
99  plot(ti,subs(z3.y,t,ti),'*k','MarkerSize',10,'LineWidth',2)
100 xlabel('t')
101 ylabel('x')
```